Ø4Ø15

I Edizione

Gianni Becattini

INTRODUZIONE ALLA PROGRAMMAZIONE

Con introduzione di F.Pirri

SISTEMI DI ELABORAZIONE - MICROPROCESSORI
VIA MONTEBELLO, 3 - 3a rosso
TEL. 055 / 219.143 — 50123 FIRENZE

INDICE

INTRODUZIONE
============

Il presente manualetto e' destinato a coloro che si avvicinano per la prima volta al linguaggio di programmazione RPN/8A. Il carattere della trattazione e' prevalentemente introduttivo; informazioni piu' dettagliate si trovano nei manuali di utenza relativi alla versione impiegata. Si e' cercato, per quanto possibile, di fare uso di un linguaggio piano ed accessibile, forse talora a scapito del rigore ma a vantaggio, almeno nelle intenzioni, della comprensibilita' da parte di persone non specificamente preparate nel settore della programmazione.

Quasi tutti gli argomenti trattati sono confortati da esempi e sono riportati anche alcuni semplici esercizi completi di soluzione.

E' in corso di preparazione un apposito volumetto contenente una piu' nutrita serie di esemplificazioni applicative con programmi di uso generale.

Si ringraziano anticipatamente i lettori per la segnalazione di eventuali errori od inesattezze che si trovassero nella presente pubblicazione.

PREMESSA

Per offrire il massimo della capacita' di calcolo, L'RPN/8A, nelle sue varie versioni, lavora con una logica coerente e naturale che puo' differire leggermente da quella di altri linguaggi. Ecco perche' questa sezione potra' risultare utile anche a chi con altri linguaggi ha gia' dimestichezza. E, naturalmente, risultera' utilissima per i principianti per cominciare ad acquistare dimestichezza con lo RPN/8A.

UN LINGUAGGIO ALLA PORTATA DI TUTTI

Da tempo si sentiva l'esigenza di un linguaggio ad alto livello adatto a funzionare sui microcomputers della serie CHILD 8/BS (x). Difatti la grande versatilita' ed il basso costo di tali apparati aveva avvicinato l'obbiettivo di realizzare un piccolo "centro di elaborazione" personale ma permaneva lo ostacolo della necessita' di programmazione in linguaggio macchina o addirittura base, spesso non gradito all'utente non specificamente preparato.

(x) Prodotti da General Processor - Firenze

Ed ecco finalmente la risposta a coloro che desiderano utilizzare il proprio CHILD 8/BS per eseguire calcoli commerciali, scientifici o statistici. Con la piu' grande facilita', paragonabile a quella con cui si apprende a far funzionare una macchina calcolatrice, chiunque puo' imparare in pochissimo tempo a scrivere i programmi per le proprie necessita'.

Tra i notevoli vantaggi dell'RPN/8A c'è quello di poter operare anche congiuntamente a telescriventi a 5 livelli in codice Baudot, molto diffuse per telecomunicazioni.

Nell'RPN/8A non si ha mai la necessità di lavorare su numeri binari o su codici a livello macchina ma e' possibile stabilire collegamenti con programmi assoluti per avere il massimo controllo di tutte le prerogative dell'elaboratore.

L'RPN/8A lavora con una precisione superiore a quella con cui sono note la maggior parte delle costanti fisiche dell'universo: 14 cifre significative piu' segno. Ed e' in grado di trattare numeri piccolissimi come 10^{-125} (e cioe' la virgola seguita da 124 zeri) fino a numeri molto elevati

come 10^{125} e cioe' 1 seguito da 126 zeri. La virgola si sposta automaticamente tenendo conto degli ordini di grandezza dei numeri trattati. La catasta operativa a "stack" e la notazione polacca inversa, usate dall'RPN/8A, sono uno dei migliori sistemi per la valutazione delle espressioni matematiche.

QUANDO SI ACCENDE IL CALCOLATORE

Quando si accende il calcolatore si danno due casi: o il sistema dispone di interprete RPN/8A su memoria ROM o meno. Nel primo caso le operazioni sono molto semplificate in quanto basta impartire da debug il comando G1000 perche' l'RPN/8A venga inizializzato e venga stampata l'intestazione con l'indicazione della versione usata e la data della medesima. Nel caso in cui non si disponga della ROM programmata, sara' necessario caricare prima in memoria l'interprete da nastro perforato o da altra periferica.

Quando si inizia l'esecuzione dell'interprete RPN/8A viene stampato a inizio riga il carattere ":" (due punti) per indicare che la macchina si attende

ordini e che e' pronta per iniziare il lavoro. Tutte le volte che il controllo torna all'operatore, ossia a chi siede alla tastiera, verranno ristampati i ":".

In alternativa al punto indicato in precedenza (H'1000'), esiste un punto di ingresso alternativo che non altera un eventuale pr-ogramma gia' esistente in memoria; corrisponde alla locazione H'1035'.

INIZIAMO CON UN ESEMPIO

Per cominciare a lavorare con l'RPN/8A possiamo provare a calcolare una approssimazione empirica del numero pi greco. A tale scopo calcoliamo il valore della frazione 355/113. Vediamo come scrivere un semplice programma per valutare tale frazione e quindi il suo scostamento dal valore di pi greco a dieci cifre significative.

Esempio;

```
* R P N / 8 A1 (Vers.26.08.77)

:[FIX] 9
:355 113/ [PRINT]
:3.141592654- 100* 3.141592654/     PRINT
:[END] (h)
:(1)[FIX] 9
355 113/ [PRINT]
3.141592654- 100* 3.141592654/ [PRINT]
[END]
:(h)
:(e)3.141592920 0.000008478
:
```

In pratica dobbiamo calcolare l'espressione (355/113 - 3.141592654) x (100/3.141592654) per vedere quanto la frazione 355/113 differisce dal valore di pi greco esatto a dieci cifre significative.

All'inizio del programma stabiliamo il numero delle cifre con cui vorremo vedere stampati i risultati a seguito dell'operatore PRINT, per mezzo dell'operatore FIX seguito dal numero 9 per visualizzare 9 cifre decimali. L'operatore FIX corrisponde nelle telescriventi ASCII al tasto "f". Si introduce poi il numero 355 semplicemente facendolo seguire da uno spazio. Si introduce quindi il numero 113 e con l'operatore di divisione / si calcola la frazione desiderata. Il risultato puo' ora essere stampato con l'operatore PRINT (tasto corrispondente in appendice). Si sottrae 3.141592654 al valore ottenuto, si moltiplica il risultato per 100 e si divide per 3.141592654. Anche questo risultato puo' essere stampato con l'operatore PRINT. Per indicare la fine del programma si usa l'operatore END.

Per verificare di aver battuto correttamente il programma, basta chiederne la "lista" ossia la stampa completa. Per fare cio' ci si deve posiziona

re ad inizio del programma stesso con il comando h (h sta per "home", cioe' "a casa", ad inizio programma) e quindi il comando l (list, lista). Per impartire un comando si batte prima il carattere ";" (punto e virgola) e la macchina risponde con la parentesi tonda aperta. Si batte il comando desiderato ed automaticamente viene richiusa la parentesi. E' facile cosi' identificare i comandi battuti sul foglio di carta. Nell'esempio sopra riportato si osserva che ad alcuni operatori (ad es. FIX, carattere "f") corrispondono sul foglio di stampa, piu' caratteri. E' la macchina che, appena battiamo un qualsiasi operatore, lo converte nella sequenza di caratteri che lo identificano. Questo vuol dire che e' sempre necessario e sufficiente battere UN SOLO carattere per ogni operatore. Tutto cio' si traduce in una velocita' di lavoro molto piu' alta. Nel programma riportato sono stati racchiusi in un riquadro le scritte corrispondenti ad un solo tasto.

Una volta verificato di non avere commesso errori, si passa nella fase di esecuzione del programma, prima tornando all'inizio col comando h e poi dando il comando e (execute, esegui). Viene cosi'

stampato il valore ora calcolato dello "pseudo pi greco" e poi l'approssimazione. Sappiamo cosi' che 355/113 approssima pi greco entro 8.47 milionesimi di 1%. Per ricordare l'approssimazione di pi greco basta scrivere le prime tre cifre dispari 113 355.

L'ARITMETICA DELL'RPN/8A

Nell'RPN/8A i risultati vengono valutati quando si incontra uno degli operatori + (somma), - (sottrazione), x (moltiplicazione), / (divisione). Come in una qualsiasi addizionatrice il tasto + somma lo ultimo dato impostato a quello che e' gia' in macchina, cosi' nell'RPN/8A gli operatori di tutte le operazioni. Pertanto per eseguire una operazione si devono PRIMA inserire gli operandi e POI l'operatore. Proviamo con qualche facile esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:12 3+ PRINT  END (h)
:(e)15.00
:(h)
:12 3 - PRINT END (h)
:(e)09.00
:(h)
:12 3* PRINT END (h)
:(e)36.00
:(h)
:12 3/ PRINT END (h)
:(e)4.00
:
```

Si noti che nei quattro programmini ora visti:

° Entrambi i numeri sono in macchina prima di effet tuare l'operazione.

° L'operatore di somma, sottrazione, moltiplicazione e divisione fa eseguire l'operazione tra i due numeri introdotti in macchina.

Nell'esempio della sottrazione abbiamo inserito uno spazio dopo il secondo numero. Cio' non provoca alcun inconveniente in quanto l'RPN/8A riconosce la "fine del numero" non appena incontra un operatore (od una variabile, vedi dopo). Il carattere "spazio" viene considerato come operatore. Come ovvio non poteva essere eliminato lo spazio tra il 12 ed il 3 in quanto sarebbe risultato il numero 123.

Esiste un particolare operatore detto PSH (push, pigia) che ricopia l'ultimo numero introdotto in un secondo registro interno (un registro non e' altro che un dispositivo che trattiene numeri). Quindi se si vuole raddoppiare un numero non occorre inserirlo una seconda volta ma basta usare il PSH e quindi l'operatore +. Lo stesso per elevarlo al quadrato: PSH e x.

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:3 PSH + PRINT END (h)
:(e)6.00
:(h)
:3 PSH  * PRINT END (h)
:(e)9.00
:
```

SEQUENZE DI OPERAZIONI

Dovendo calcolare manualmente ((2+3)/4+5)x6 si somma prima 2 e 3, si divide la somma per 4, si somma 5 al risultato ed infine si moltiplica per 6. Questo e' esattamente cio' che si fa con l'RPN/8A:

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:2 3+ 4/ 5+ 6* PRINT END (h)
:(e)37.50
:
```

SOMME DI PRODOTTI

Facciamo un esempio elementare. Supponiamo di avere venduto 12 articoli al prezzo di lire 920 ciascuno, 8 a 1550 e 16 a 315. L'incasso totale vale:

(12x920)+(8x1550)+(16x315)

Possiamo calcolarne facilmente il risultato

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:12 920*, 8 1550*+, 16 315*+
:"Il valore dell'incasso e' " PRINT
:END (h)
:(e)Il valore dell'incasso e' 28480.00
:
```

Ovviamente in questo modo possiamo trovare la somma di un numero qualsivoglia di prodotti. Notiamo che in questo esempio abbiamo aggiunto l'operatore TEXT, costituito dal carattere " (doppio apice). Grazie al TEXT possiamo far stampare, in fase di esecuzione, delle frasi a piacere, semplicemente racchiudendole, nel programma, tra doppi apici. E' stato cosi' possibile aggiungere la scritta esplicativa Il valore dell'incasso e' davanti alla stampa del risultato. Altra novita' e' costituita dall'uso del carattere virgola (operatore SEPARAZIONE) il cui scopo e' quello di conferire una migliore chiarezza grafica al programma.

L'uso dell'operatore TEXT rende molto piu' comunicativa l'esecuzione del programma come chiariranno meglio gli esempi piu' complessi.

PRODOTTI DI SOMME
=================

Problemi del tipo (7+3)x(5+11)x(13+17) si risol vono nello stesso modo ma scambiando tra loro gli o- peratori + e x.

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:7 3 +, 5 11+*, 13 17+* PRINT END (h)
:(e)4800.00
:
```

LA CATASTA DEI REGISTRI OPERATIVI
=================================

Negli ultimi esempi l'RPN/8A ha doyuto mettere da parte alcuni risultati parziali per usarli suc- cessivamente. Vediamo come puo' farlo. Ci sono nel- l'RPN/8A quattro registri per dati che chiameremo X, Y, Z, T. Essi sono sistemati in modo da formare una catasta (in inglese "stack") con X al fondo e T in cima. Per evitare possibili confusioni tra le denominazioni dei registri ed i dati in essi con- tenuti, indicheremo i registri con lettere MAIUSCO LE ed i dati con lettere minuscole. Cosi' x, y, z, t, rappresentano i dati contenuti in X, Y, Z, T. Quando si imposta un numero esso viene automatica-

mente inserito, in fase di esecuzione, nel registro X. Se impostiamo poi un altro numero, il numero precedentemente introdotto passa in Y, tutta la catasta slitta verso l'alto di un posto (il contenuto del registro T va perduto) ed il nuovo numero viene posto in X. L'operatore PSH invece "ricopia" il contenuto di X in Y ed anche in questo caso la catasta sale di un posto. Quando viene eseguito l'operatore di somma (+), x va a sommarsi ad y ed il risultato vine posto in X, mentre la catasta scende di un posto. In T rimane x. Lo stesso accade per gli operatori di sottrazione, moltiplicazione, divisione.

L'operatore STK serve per provocare, durante l'esecuzione di un programma, la stampa di tutti i registri dello stack. Ce ne serviremo nello esempio seguente per verificare cosa accade nello stack ogni volta che si effettuano delle operazioni.

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:3 STK $
:4 STK $
:* STK $
:5 STK $
:6 STK $
:* STK $
:+ STK $
:"Valore dell'operazione " PRINT
:$ STK
:END (h)
:
```

```
(h)
:(1)3 STK $
4 STK $
* STK $
5 STK $
6 STK $
* STK $
+ STK $
"Valore dell'operazione " PRINT
$ STK
END
:(h)
:(e)
X: 3.00
Y: .00
Z: .00
T: .00

X: 4.00
Y: 3.00
Z: .00
T: .00

X: 12.00
Y: .00
Z: .00
T: 4.00

X: 5.00
Y: 12.00
Z: .00
T: .00

X: 6.00
Y: 5.00
Z: 12.00
T: .00

X: 30.00
Y: 12.00
Z: .00
T: 6.00

X: 42.00
Y: .00
Z: 6.00
T: 30.00
Valore dell'operazione 42.00

X: 42.00
Y: .00
Z: 6.00
T: 30.00
:
```

Osserviamo alcune cose importanti:

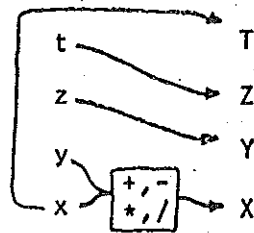
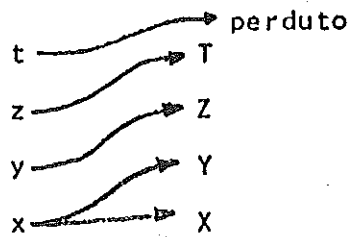
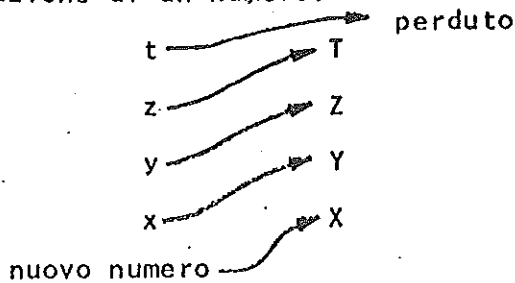
° Abbiamo fatto la conosc enza con un nuovo operatore, il $, che ha l'effetto di far tornare a capo, a rigo nuovo, il carrello della stampante.
° Dopo una operazione che ha fatto scendere lo stack rimane in T il vecchio contenuto di X, che prosegue poi verso il basso in caso di ulteriori "discese".
° L'esecuzione dell'operatore PRINT non ha modificato lo stack, e cosi' la stampa della frase con l'operatore TEXT.

Le operazioni monadiche, ossia con un solo argomento, come la radice quadrata, il seno ecc., lavorano sempre sul registro X.

A pagina seguente sono riportate alcune figure illustrative inerenti lo stack.

NUMERI NEGATIVI

E' possibile introdurre nello stack anche numeri negativi, semplicemente cambiando il segno del registro X una volta introdotto il numero. L'operatore destinato a tale funzione e' il CHS (CHange

t
registro T
z
registro Z
y
registro Y
x
registro X
Operazioni:
t
z
y
x
T
Z
Y
X
+, -
*, /
Push:
perduto
t
z
y
x
T
Z
Y
X

Introduzione di un numero:
perduto
t
z
y
x
T
Z
Y
X
nuovo numero

Sign, ossia, cambia il segno). Quando si introduce un numero, il segno del registro X viene automaticamente fissato come positivo, e si deve usare l'operatore CHS per farlo diventare negativo. Al termine di una operazione invece il registro X puo' contenere anche un numero negativo. In tal caso il CHS cambia numeri positivi in negativi e negativi in positivi.

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:32 4* PRINT CHS PRINT END (h)
:(e)128.00 -128.00



:(h)
:21 CHS 4*PRINT CHS PRINT STO \END (h)
:(e)-84.00 84.00
:
```

Si noti che nel secondo esempio abbiamo battuto per errore il tasto corrispondente all'operatore STO invece di quello corrispondente all'operatore END. Per eliminarlo dalla memoria abbiamo usato il tasto di CANCELLAZIONE che stampa una "\" tutte le volte che viene premuto e che cancella tanti caratteri quante sono le volte che e' stato premuto. Dal

punto di vista di occupazione della memoria, E QUESTO E' MOLTO IMPORTANTE, anche gli operatori che vengono stampati come piu' caratteri, ad es. PRINT, occupano una sola locazione di memoria. Per cancellare STO (esempio precedente) si batte cosi' una sola volta il tasto di cancellazione.

NUMERI DECIMALI

Quando si batte il programma si possono anche introdurre numeri aventi parte decimale senza nessun particolare accorgimento. Non e' ammessa la notazione .034, ossia non si possono iniziare i numeri per il carattere "." ma si scrivera' 0.034. I numeri aventi piu' di 14 cifre a sinistra del punto decimale possono essere introdotti solo in formato esponenziale (e solo qualora la versione di RPN/8A ne disponga della possibilita'). Gli zeri non significativi, tanto prima che dopo il punto decimale, sono ignorati.

Dal punto di vista invece della stampa valgono le seguenti considerazioni. Prima di effettuare una

stampa bisogna informare la macchina del numero di decimali che vogliamo vedere stampati con l'operatore FIX cui abbiamo gia' fatto cenno nella introduzione. L'operatore FIX deve essere seguito, SENZA AGGIUNGERE SPAZI oltre quello che la macchina introduce automaticamente dopo ogni operatore che richieda piu' di un carattere di stampa, dal numero dei decimali che si desiderano visualizzare. Oltre i 9 decimali, corrispondenti da quanto detto al numero 9,si devono usare le seguenti lettere:

10 decimali: lettera J
11 " " K
12 " " L
13 " " M
14 " " N

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:1 3/ PRINT FIX K PRINT END (h)
:(e)0.33 0.33333333333
:
```

Come si osserva anche dall'esempio ora visto, viene

automaticamente assunto il formato FIX 2 qualora non ne venga specificato uno differente.

Con l'operatore PRINT non si possono stampare numeri aventi piu' di 14 cifre a sinistra del punto decimale e questo nel formato FIX 0. Nel FIX 1 tali cifre sono ridotte a 13, a 12 nel FIX 2 e cosi' via.

Numeri maggiori devono essere stampati in formato esponenziale.

REGISTRI DI DEPOSITO: LE VARIABILI

Nell'RPN/8A ci sono 16 registri di memoria che possono servire per conservarci delle costanti. Ciascuno di tali registri si chiama VARIABILE e si identifica con una delle prime 15 lettere dell'alfabeto inglese o col carattere"@" (tale carattere può differire a seconda del tipo di telescrivente usata). Le variabili non sono di regola alterate dalla esecuzione dei calcoli nella catasta operativa. Solo talora, nel caso di funzioni piu' complesse, possono essere alterate da alcuni operatori (nel qual caso ne e' data esplicita informazione nel manuale di utenza). Per introdurre un numero in una variabile

bisogna far precedere il nome della variabile dallo operatore STO (STOre, immagazzina). In questo modo il contenuto del registro X viene ricopiato nella variabile indicata. Per richiamare una variabile ed introdurla nel registro X basta scriverne il nome.

Esempio:

```
* R P N / 8 A1 (Vers.28.08.77)

:1(h)
:(1)1 2 3 4 STK $
STO A
STK $
5 6 7 8 STK $
A 21* PRINT
$ STK
END
:(h)
:(e)
X: 4.00
Y: 3.00
Z: 2.00
T: 1.00

X: 4.00
Y: 3.00
Z: 2.00
T: 1.00

X: 8.00
Y: 7.00
Z: 6.00
T: 5.00
84.00

X: 84.00
Y: 8.00
Z: 7.00
T: 21.00
:
```

Osservazioni:

° L'esecuzione dell'operatore STO non altera lo stack

° Quando si richiama una variabile il numero in essa contenuto viene ricopiato nel registro X e lo stack sale di un posto.

° Quando si fa una operazione di richiamo di una variabile, il numero in essa contenuto non viene alterato.

ELABORAZIONE INTERATTIVA: RICHIESTA DATI DA TASTIERA

Per avere una effettiva versatilita' di uso, l'RPN/ /8A consente di interagire con il programma durante la esecuzione stessa per alterarne lo svolgimento o per introdurre dati. A tale fine si fa uso dell'operatore INPUT. Quando si incontra l'operatore INPUT l'elaborazione viene temporaneamente sospesa e la telescrivente emette un suono della campana (bell). L'operatore puo' quindi battere un numero dalla tastiera. Detto numero puo' essere battuto negli stessi formati indicati in precedenza (vedi paragrafo "NUMERI DECIMALI") ma non puo' essere composto da piu' di 30 caratteri. Per far

riprendere l'esecuzione ci sono due alternative secondo che il numero sia positivo o negativo: nel primo caso si batte il "Ritorno carrello", nel secondo il tasto"-". In tale istante il numero battuto viene introdotto nel registro X e lo stack sale di un posto. Se si commettono degli errori, prima di riprendere la elaborazione, si puo' usare il tasto cancellazione come spiegato in precedenza.

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:"Batti un numero " INPUT 3* $ PRINT END (h)
:(e)Batti un numero 457.2
1371.60
:(h)
:(e)Batti un numero 4-
-12.00
:(h)
:(e)Batti un numero 45\4
132.00
:
```

COME MANIPOLARE LA CATASTA

Abbiamo a disposizione tre operatori per riorganizzare la catasta operativa.:

RD - (Roll Down, ruota verso il basso)

RU - (Roll Up, ruota verso l'alto)

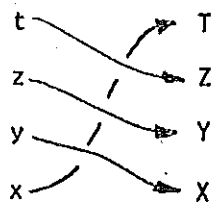
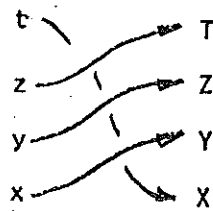
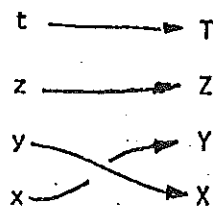
X()I - (X scambio con Y)

Come intuitivo il RD fa scorrere lo stack di un posto verso il basso (vedi figure illustrative a pagina seguente) mentre RU fa il contrario. L'operatore X()Y invece fa si' che i registri X ed Y si scambino i contenuti. Cio' e' spesso utile prima di sottrazioni o divisioni.

Esempio:

```
:11 22 33 44 $ STK
:RD $ STK
:RU $ STK
:X()Y $ STK
:END :(h)
:(e)

X: 44.00
Y: 33.00
Z: 22.00
T: 11.00

X: 33.00
Y: 22.00
Z: 11.00
T: 44.00

X: 44.00
Y: 33.00
Z: 22.00
T: 11.00

X: 33.00
Y: 44.00
Z: 22.00
T: 11.00
:
```

RD:

t → T
z → Z
y → Y
x → X

RU:

X()Y:

Osservazione:

° Per recuperare l'ultimo x dopo una operazione (quel lo che in molti calcolatori si chiama "Last x") basta eseguire un RU (Roll Up).

QUANDO UN PROGRAMMA ARRIVA ALLA FINE

Ogni programma RPN/8A DEVE terminare con lo operatore END. Non appena, durante l'esecuzione, es so viene incontrato il controllo torna all'operatore e la macchina stampa i due punti in attesa di ordini.

Quando il programma e' composto da diversi rami si puo' usare, per arrestare l'esecuzione, anche l'operatore STOP, il cui effetto e' del tutto simile a quello dell'END. La differenza e' che l'operatore END serve anche per segnalare la fine fisica del programma (per esempio al termine della funzione di list), mentre lo stop segnala solo il termine della esecuzione. Cosi' potremo avere nel programma diversi operatori STOP ma un solo END. Una delle applicazioni dell'operatore STOP e' quella di consentire la memorizzazione contemporanea di piu' programmi e la

esecuzione di essi uno alla volta, a piacere. Questo verra' esaminato meglio nel paragrafo successivo.

PIU' PROGRAMMI IN MEMORIA

E' possibile inserire contemporaneamente in memoria piu' di un programma RPN/8A ed eseguirne uno alla volta. E' necessario identificare in qualche modo i diversi programmi e pertanto si munisce ciascuno di essi con una etichetta (in inglese LABEL, da cui il nome dell'operatore LBL). Una etichetta non e' altro che un modo di identificare un punto del programma per potervi fare riferimento. Vediamo come si compone una etichetta: per prima cosa si deve porre l'operatore LBL, che viene automaticamente seguito da uno spazio. Indi, senza aggiungere spazi intermedi, si deve porre un codice a piacere. Ad esempio, sono etichette valide LBL 7 oppure LBL A. E' ammesso anche l'uso di operatori, come etichette, ad es LBL STO. In questo caso STO non ha piu' significato di assegnazione a variabile bensi' quello di simbolo. E' comunque preferibile usare i numeri e le lettere, e ricorrere ai nomi di operatori solo in caso di necessità.

Le lettere usabili direttamente sono quelle delle variabili: A...O ed a.

Per comprendere come si puo' passare alla ese cuzione di un determinato programma e' necessario chiarire prima la funzione e l'uso del comando f (find, trova). Abbiamo visto come prima di listare od eseguire un programma ci si"portasse all'inizio" del medesimo tramite il comando h (home). Avevamo in effetti voluto dare una indicazione intuitiva. In pratica esiste un puntatore (pointer), ossia un indice mobile,che puo' essere indirizzato su qualunque locazione della memoria programma. Con il comando h si porta il puntatore sulla PRIMA di dette locazioni e, poiche' il nostro programma i- niziava dalla prima locazione si poteva pas sare in esecuzione. Con il comando f invece si puo' localizzare il pointer su una qualsiasi locazione semplicemente fornendo i due elementi che la pre- cedono. Ad esempio, il comando(f)LBL 5 indirizza sulla locazione subito seguente LBL 5, ammesso ov- viamente che la sequenza LBL 5 esista all'interno del programma e che la ricerca sia stata eseguita in maniera corretta. Infatti la ricerca INIZIA dalla

locazione cui punta il pointer quando viene imparti to il comando f e TERMINA alla END. Al termine della ricerca, se questa ha avuto esito negativo, viene stampata la scritta NOT FND (NOT FouND, non trovato); in caso positivo invece vengono stampati i due punti per indicare la corretta posizionatura del pointer. Questo modo di operare risulta utile quando si debba no ricercare due caratteri ripetuti piu' volte nel corso di un programma.

Per eseguire un programma tra diversi che si trovano in memoria si dovra' quindi:

1) Dare il comando h (home)
2) Dare il comando f (find) seguito da LBL ed il codice del programma desiderato (Es. (f)LBL 5)
3) Quando la macchina stampa : dare il comando e (execute)

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:LBL 1 "Programma 1" $ 3 4* PRINT STOP
:LBL 2 "Programma 2" $
:5 6+ PRINT STOP
:LBL 3 "Programma 3" STOP
:STO \END (h)
:(1)LBL 1 "Programma 1" $ 3 4* PRINT STOP
LBL 2 "Programma 2" $
5 6+ PRINT STOP
LBL 3 "Programma 3" STOP
END
:(h)
:(f)LBL 2
:(e)Programma 2
11.00
```

```
:(h)
:(f)LBL 3
:(e)Programma 3
:(h)
:(f)LBL 1
:(e)Programma 1
12:00
:.
```

Osservazione:

° Dopo una ricerca con esito negativo il pointer viene riportato in posizione home (prima locazione memoria programma)

PROGRAMMAZIONE RIPETITIVA

I programmi visti finora sono tutti di tipo non ripetitivo, ossia vengono eseguiti una sola volta prima della fine. Uno dei maggiori vantaggi di un elaboratore consiste proprio nel poter eseguire delle procedure per un numero elevato di volte sotto il controllo automatico dell'elaboratore stesso.

Per introdurre il concetto di ripetitivita' ricorreremo ad un esempio nel corso del quale faremo uso di due nuovi operatori:

GOTO - Significa "vai a" e passa il controllo al punto del programma identificato alla etichetta specificata. Ad es. GOTO 3 passa il controllo alla locazione successiva a LBL 3. Tra GOTO e codice non devono esserci spazi oltre quello automaticamente inserito dalla macchina.

IF X.LT.0 - Significa SE X E' MINORE DI ZERO (LT sta per Less Than, minore di). La sua funzione e' la seguente:
Se il contenuto di X e' minore di zero l'esecuzione continua come se niente fosse.
Se il contenuto di X non e' minore di zero la esecuzione procede dalla linea di programma subito successiva.
Esempio: Una maniera di calcolare la radice quadrata di un numero e' il seguente:

1) Si ipotizza di conoscere la radice quadrata del numero stesso; sia tale numero B.
2) Si valuta l'espressione

(A/B+B) / 2 (1)

dove A e' il numero del quale vogliamo trovare la radice. Il risultato viene assegnato a B.
3) Si valuta B^2

4) Si fa la differenza A-B e se ne considera il modulo (valore assoluto, ossia il numero senza il segno)

5) Tale differenza, che e' la differenza tra il numero di cui vogliamo trovare la radice ed il quadrato della radice che noi abbiamo trovato, e' sufficientemente piccola? Vale a dire, e' minore dell'errore massimo che noi possiamo tollerare?

6) Se si, il lavoro e' finito e B rappresenta il valore della radice cercata. Se no, si torna al passo numero 2.

Nel programma esempio che segue, il numero B viene inizialmente posto uguale ad A e vengono contate le iterazioni compiute, ossia quante volte viene eseguito il ciclo, prima di ottenere la approssimazione richiesta (una parte su 10^{-7} , massimo scostamento 0.0000001).
Per motivo di spazio tale programma e' riportato nella pagina seguente.

Il numero delle iterazioni viene stampato in formato FIX 9. Sarebbe stato piu' estetico modificarlo in FIX 1 come il lettore puo' verificare per esercizio.

```
:(h)
:(1)FIX 9. $. "Numero da cui estrarre la radice? " INPUT . STO A. STO B
0 STO C
LBL 1 A B/. B+. 2/ STO B
C 1+ STO C
A B PSH *- ABS . 0.0000001-
IF X.LT.0 $" Radice cercata= " B PRINT . " Numero Iterazioni= " C PRINT STOP
GOTO 1
END
:(h)
:(e)
Numero da cui estrarre la radice? 625.2
 Radice cercata= 25.003999680  Numero Iterazioni= 9.000000000
:(h)
:(e)
Numero da cui estrarre la radice? 0.000789
 Radice cercata= 0.028089175  Numero Iterazioni= 8.000000000
:(h)
```

Osservazioni:

° L'operatore ABS che non avevamo ancora incontrato calcola il valore assoluto (o modulo) del contenuto del registro X e mette il risultato in X. Ad es il modulo di 4 e' 4, il modulo di -5 e' 5.

I SOTTOPROGRAMMI

I sottoprogrammi sono delle speciali sezioni di programma che possono essere richiamati anche piu' volte da altri programmi. Ad esempio, se dobbiamo scrivere un programma che richieda di effettuare piu' volte il calcolo della radice quadrata sarebbe troppo scomodo ed antieconomico ai fini della occupazione di memoria riscrivere tutte le volte il programma visto sopra. E' molto piu' conveniente organizzare il calcolo della radice quadrata come sottoprogramma da richiamare tutte le volte che sia necessario.

Un sottoprogramma inizia con una etichetta, in modo da poterlo richiamare, e termina con uno operatore RETURN che fa ritornare l'esecuzione al punto subito dopo quello in cui il sottoprogramma era stato chiamato.

Il programma che richiama il sottoprogramma si chiama PRINCIPALE (in inglese, main) o CHIAMANTE. I sottoprogrammi sono detti anche SUBROUTINES. Per richiamarli si usa l'operatore CALL (chiama).
Esempio:

```
:(1)25 CALL 1 PRINT , 625 CALL 1 PRINT STOP
LBL 1 STO A STO B
LBL 2, A B/ B+ 2/ STO B
A B B *- ABS 0.00000001-
IF X.LT.0 B RETURN
GOTO 2
END
:(h)
:(e)5.00 25.00
:
```

NOTA IMPORTANTE: Le varie versioni di RPN/8A trat_ tano le subroutines in maniera piuttosto differente: Esistono versioni in cui una subroutine puo' a sua volta chiamarne un altra, nessuna o addirittura ave- re alcune limitazioni sugli operatori usabili in seno al sottoprogramma. E' questo il caso dello e- sempio precedente dove l'operatore PSH e' stato so- stituito dalla ripetizione della variabile B.
E' conveniente riferirsi sempre al manuale della versione in uso.

ALTRI OPERATORI DI CONTROLLO E DECISIONE

Oltre al gia' visto operatore IF X.LT.0 ne esi_ stono altri per il controllo e decisione. Vediamo i piu' comuni.

Con IF X.EQ.0 (if x equal zero,ossia, se x e' uguale a zero) si controlla il valore di x e se risulta differente da zero si salta a nuova riga. In IF X.EQ.Y il salto avviene solo se i contenuti dei registri X ed Y sono diversi. Combinando questi operatori e' possibile ottenere quasi ogni tipo di controllo su valori numerici.

Nell'esempio che segue e' riportata una semplice maniera di controllare la risposta dell'operatore ad una domanda. Si fa la convenzione che 1 stia per SI e qualunque altra cosa per NO.

Esempio

```
* R P N / 8 A1 (Vers.26.08.77)

:";h"(h)
:$ VAR (h)
:$ "Vuoi una riga? " INPUT
:1 IF X.EQ.Y "*****************"
:END (h)
:(e)
Vuoi una riga? 1*****************
:(h)
:(e)
Vuoi una riga? 0
:
```

TRACCIARE GRAFICI CON L'RPN/8A

Grazie agli operatori dell'RPN/8A e' possibile anche tracciare dei semplici grafici con l'ausilio della telescrivente. E' possibile tracciare diagrammi, istogrammi, tabulare funzioni, eseguire semplici disegni.

L'operatore che sta alla base del tracciamento di grafici e' il TAB che produce sul foglio della stampante tanti spazi tanto vale il numero contenuto nel registro X senza riguardo al segno. Se il numero e' frazionario (ossia con parte decimale diversa da zero) si effettua l'arrotondamento allo intero superiore (7.876 provoca 8 spazi, cosi' 7.008).
Gli spazi vengono eseguiti dalla posizione in cui si trova il carrello della stampante quando si esegue l'operatore TAB.

Vediamo un esempio abbastanza completo in cui sono riunite molte delle nozioni fin qui esposte.
Il programma traccia il grafico della funzione X^3-X^2 con possibilita' di introdurre da tastiera i vari parametri per esaminare la funzione come si desidera.

Esempio:

```
* R P N / 8 A1 (Vers.26.08.77)

:FIX 3
:$$$"              T A B U L A Z I O N E  D I  X**3-X**2".$$
:$
:LBL 3. "Estremi dell'intervallo (B>A)"$"A? " INPUT STO A
:" B? " INPUT STO B $
:"Intervallo di campionamento? " INPUT STO I $
:"Fattore di scala? " INPUT STO F $$$
:A STO C
:LBL 1 CC*.C*.CC*- STO D F* 35+ TAB "* " D PRINT $
:B C- IF X.LT.0 GOTO 2
:C I+ STO C GOTO 1
:LBL 2 $"Ancora? " INPUT 1 IF X.EQ.Y GOTO 3
:END (h)
:(1)FIX 3
$$$"              T A B U L A Z I O N E  D I  X**3-X**2".$$
$
LBL 3. "Estremi dell'intervallo (B>A)"$"A? " INPUT STO A
" B? " INPUT STO B $
"Intervallo di campionamento? " INPUT STO I $
"Fattore di scala? " INPUT STO F $$$
A STO C
LBL 1 CC*.C*.CC*- STO D F* 35+ TAB "* " D PRINT $
B C- IF X.LT.0 GOTO 2
C I+ STO C GOTO 1
LBL 2 $"Ancora? " INPUT 1 IF X.EQ.Y GOTO 3
END
:(h)
:(e)
```

T A B U L A Z I O N E D I X**3-X**2

```
Estremi dell'intervallo (B>A)
A? 2.9- B? 3
Intervallo di campionamento? 0.2
Fattore di scala? 1


 * -32.799
       * -26.973
            * -21.875
                * -17.457
                    * -13.671
                       * -10.469
                          * -7.803
                            * -5.625
                              * -3.887
                               * -2.541
                                * -1.539
                                 * -0.833
                                 * -0.375
                                 * -0.117
                                 * -0.011
                                 * -0.009
                                 * -0.063
                                 * -0.125
                                 * -0.147
                                 * -0.081
                                  * 0.121
                                  * 0.507
                                   * 1.125
                                    * 2.023
                                     * 3.249
                                      * 4.851
                                        * 06.877
                                           * 09.375
                                              * 12.393
                                                 * 15.979
                                                      * 20.181

Ancora? 1
```

T A B U L A Z I O N E   D I   X**3-X**2

```
Estremi dell'intervallo (B>A)
A? 0.5- B? 1
Intervallo di campionamento? 0.05
Fattore di scala? 100

* -0.375
   * -0.293
          * -0.224
                * -0.165
                     * -0.117
                         * -0.078
                            * -0.048
                              * -0.025
                               * -0.011
                                * -0.002
                                * -0.000
                                * -0.002
                                * -0.009
                               * -0.019
                             * -0.032
                           * -0.046
                         * -0.063
                        * -0.079
                      * -0.096
                    * -0.111
                   * -0.125
                  * -0.136
                 * -0.144
                 * -0.147
                 * -0.147
                 * -0.140
                   * -0.128
                     * -0.108
                       * -0.081
                           * -0.045
                               * -0.000
```

Nel primo esempio abbiamo fatto tabulare la funzione in un ampio intervallo attorno allo zero. Successivamente, incuriositi dall'andamento dei valori in prossimita' dell'origine, abbiamo fatto espandere la scala di un fattore 100 riducendo gli estremi di osservazione e l'intervallo di campionamento. Abbiamo in tal modo ottenuto il diagramma di pagina precedente dove si puo' infatti osservare una oscillazione ad un piccolo valore negativo.

Complicando ancora un po' il programma si potrebbe far tracciare anche gli assi, i valori delle ascisse ecc.

ULTERIORE POTENZA PER GLI ESPERTI

Chi ha buona conosc enza di programmazione o assembler, potra' egli stesso scrivere dei programmi in linguaggio base cui fare riferimento da un programma RPN/8A. Si comprende come in tal modo si espanda il campo di applicazione dell'RPN/8A: e' possibile interfacciare quasi ogni tipo di periferiche o sistemi da controllare scrivendo i relativi drivers, conservando tutte le prerogative dell'inter

prete RPN/8A. Per fare cio' si usano gli operatori JMP e USER. Il primo ha la funzione di provocare il salto (JuMP, salta) alla locazione specificata dalle prime 4 cifre del registro X. Per esempio, per saltare alla locazione H'3756' basta scrivere 3756 JMP. L'operatore USER e' analogo, ma l'indirizzo di salto e' fisso alla locazione H'00BF' dove l'utente potra' allocare un'ulteriore istruzione di salto (questa volta in linguaggio base) ottenendo cosi' un salto indiretto. Il ritorno avviene con un salto alla locazione H'12F0'.

I dati possono essere trasferiti tra i programmi base ed RPN/8 e viceversa tramite il registro I. Il registro I e' uno speciale magazzino per dati dalle molte applicazioni il cui indirizzo assoluto e' H'00B6' e che e' collegato al registro X da un operatore X()I (X scambio con I). Ogni volta che lo si incontra nel corso della esecuzione di un programma i contenuti dei registri X ed I vengono scambiati tra di loro. Ecco come e' disposto il dato nel registro I:

punto decimale ideale

LOCAZIONI DI MEMORIA: B6 . B7 B8 B9 BA BB BC BD BE

Nella locazione B6 sta il segno: H'00' indica un numero positivo, H'F0' indica un numero negativo. Nelle locazioni B7 (msd) - BD (lsd) sta la mantissa nella forma 0.XXXXXXXXXXXXXX, essendo ogni cifra rappresentata da mezza locazione di memoria in codice BCD. Nella locazione BE sta l'esponente in formato binario. Ad H'03' per esempio corrisponde un esponente 10^3 per cui moltiplicare la mantissa. Se la mantissa fosse ½ 12345678901234 e l'esponente H'03' si avrebbe il numero decimale 123.45678901234.

E' possibile pertanto stabilire un collegamento anche con periferiche molto complesse come macchine operatrici, sistemi di misura e di controllo ecc.

ALCUNI ESERCIZI SULLA NOTAZIONE POLACCA INVERSA

Il lettore che lo desideri potra' cimentarsi nella risoluzione di alcuni semplici esercizi sulla notazione polacca inversa per chiarirne il modo di uso. Nessuno degli esercizi richiede ripetute impostazioni di dati.

Esercizio 1 - Scrivere un programma che risolva le seguenti espressioni:

1) (3x4)+(5x6)+(7x9)

2) (3+4)x(5+6)x(7+9)

3) $\left(\frac{4x5}{7} + \frac{29}{3x11}\right)\left(\frac{19}{2+4} + \frac{13+3.1415}{4}\right)$

4) $\frac{1}{\frac{1}{3}+\frac{1}{6}}$

5) $3+\frac{1}{7+\frac{1}{15+\frac{1}{1+\frac{1}{292}}}}$

Esercizio 2 - Scrivere un programma che richieda da tastiera una misura in pollici ed in piedi e la converta in centimetri. Si ricorda che 1"(pollice) = = 12' (piedi) = 2.54 cm

Esercizio 3 - Calcolare il seno di 43° usando la formula: sin(x)= 0.01 $(((0.005924\ x)^2 - 1.257)^2 + 1.645)x$

Risoluzioni: vedi pagina successiva.

RISOLUZIONI DEGLI ESERCIZI PRECEDENTI

Esercizio 1

```
* R P N / 8 A1 (Vers.26.08.77)

:3 4* 5 6*+ 7 9*+ PRINT END (h)  ]  (N°1)
:(e)105.00
:(h)
:3 4+ 5 6+* 7 9+* PRINT END (h)  ]  (N°2)
:(e)1232.00
:(h)
:4 5*7/29 3/11/+19 2 4+/13 3.1415+4/+* PRINT END (h)](N°3)
:(e)26.90
:(h)
:1 3/ 1 6/+ 1X()Y / PRINT END (h)] (N°4)
:(e)2.00


:(1)FIX 9 1 292/ 1+ 1 X()Y / 15 + 1 X()Y / 7+ 1 X()Y / 3+ PRINT END]
:(h)
:(e)3.141592653        Si noti l'approssimazione a pi greco      (N°5)
:(h)
```

Esercizio N°2

```
* R P N / 8 A1 (Vers.26.08.77)

:$(h)
:(1)$ "Quanti piedi? " INPUT 12*
$ "Quanti pollici? " INPUT +
$ "La misura indicata equivale a cm " 2.54* PRINT
END
:(h)
```

```
:(e)
Quanti piedi? 5
Quanti pollici? 3
La misura indicata equivale a cm 160.02
:(h)
:(e)
Quanti piedi? 0
Quanti pollici? 37
La misura indicata equivale a cm 93.98
:
```

Esercizio N°3

```
(h)
:(1)FIX 3
43 STO A
0.005924 A* PSH * 1.257- PSH * 0.1645+ 0.01* A* PRINT END
:(h)
:(e).681
:
```

Come ovvio gli esercizi piu' complessi possono essere risolti in una infinita' di maniere. In linea generale esistono due concetti di "migliore dei modi": il primo ai fini della utilizzazione di memoria, il secondo ai fini della velocita' di esecuzione.
E' in corso di preparazione il manualetto contenente una raccolta di esercizi e programmi applicativi di uso generale.

COME SI MODIFICA UN PROGRAMMA

Non e' detto naturalmente che le cose debbano sempre filare lisce; esisteranno molti casi in cui l'operatore si accorgera' di aver commesso qualche sbaglio nella stesura di un programma o piu' semplicemente in cui egli voglia modificarlo per adattarlo a nuove esigenze. Vediamo come si opera in tali circostanze. Per prima cosa e' necessario sapere come spostare il puntatore sulle varie locazioni della memoria programma. Abbiamo gia' visto i comandi h (home) e f (find). Ce ne sono anche altri studiati per una piu' facile correzione dei programmi (questo processo si chiama normalmente "editing"). Vediamo quali essi sono:

b - (Beginning of line, inizio della linea) - Riporta il puntatore sulla prima locazione della riga in corso. E' molto utile per riiniziarne la battitura. Puo' servire quindi come "cancella riga in corso di battitura".

g - (Give) - Fa stampare il contenuto della locazio

ne in quel momento indirizzata dal pointer. Serve per individuare la posizione di quest'ultimo.

d - (Decrement) - Sposta il puntatore indietro di una posizione e stampa l'elemento su cui si posiziona.

n - (Next, successivo) - Sposta il puntatore avanti di una locazione e stampa l'elemento su cui si posiziona.

Oltre a quelli ora detti, l'RPN/8A, almeno in certe versioni, dispone di due importanti comandi: uno per inserire un nuovo elemento di programma (comando i, insert, inserisci) ed uno per cancellare un elemento gia' esistente (comando k, kill, sopprimi) Entrabi RIORGANIZZANO AUTOMATICAMENTE il programma in memoria, ossia, nel caso del comando insert tutti gli elementi successivi vengono fatti scorrere verso il basso e nel caso del comando kill vengono fatti risalire in modo da non lasciare spazi vuoti. Vediamo come si usano:

i - insert - Si posiziona il puntatore sull'elemento PRIMA del quale si desidera fare l'inserzione. Si

impartisce quindi il comando "i" seguito da cio' che si vuole inserire. Si puo' ripetere il comando per inserire piu' elementi.

k - kill -Si posiziona il puntatore sull'elemento da eliminare e si impartisce il comando "k".

Vediamo di chiarire quanto esposto con un esempio:

```
* R P N / 8 A1 (Vers.28.08.77)

:25 CALL 1 PRINT ; 625 CALL 1 PRINT STOP
:PRINT  $ STO A (b)PRINT
:LBL 1 STO A STO B
:LBL 2; A B/ B+ 2/ STO B
:A B B *- 0.000000001-
:END (h)
:(f)*-
:(i)ABS
:(h)
:(l)25 CALL 1 PRINT ; 625 CALL 1 PRINT STOP
LBL 1 STO A STO B
LBL 2; A B/ B+ 2/ STO B
A B B *-ABS  0.000000001-
END
:(h)
:(f)*-
:(k)
:(l) 0.000000001-
END
:(h)
:(l)25 CALL 1 PRINT ; 625 CALL 1 PRINT STOP
LBL 1 STO A STO B
LBL 2; A B/ B+ 2/ STO B
A B B *- 0.000000001-
```

```
END
:(h)
:(f)B+
:(d)+
:(d)B
:(n)+
:(n)
:(n)2

:(h)
:(f)CALL 1
:(g)
:(n)PRINT
:(g)PRINT
:(n),
:(g);
:
```

COSA FARE QUANDO SIAMO NEI GUAI

Anche nei sistemi forniti di memoria ROM, nei quali non esiste la possibilita' di autocancellazione da parte del programma interpretatore RPN/8A a seguito di qualche errore di programmazione, ci possono essere dei casi in cui non risulta possibile tornare sotto lo stato di INPUT quando la macchina stampa i due punti. Potrebbe ad esempio trattarsi di un ciclo ripetuto indefinitamente; una situazione

insomma in cui si desidera interrompere l'esecuzione del programma. Le versioni piu' avanzate di RPN/8A dispongono di uno speciale tasto destinato proprio a tale scopo: basta premerlo per tornare sotto lo stato di INPUT. Nelle altre versioni e' invece necessario tornare in debug con lo switch RESET sul pannello del microcalcolatore ed impartire quindi il comando G1000 se si desidera iniziare da capo tutto il lavoro, oppure G1035 per conservare il programma in memoria.

LIMITAZIONI OPERATIVE

La precisione dell'RPN/8A dipende dalla operazione svolta. Operazioni elementari come somma, sottrazione, moltiplicazione, divisione hanno un errore massimo di ±1 nella quattordicesima cifra meno significativa. Gli errori in queste operazioni sono dettati dalla necessita' di troncare i risultati. Per esempio, $(\frac{1}{3}) \times 3$ non da' come risultato 1, come sarebbe giusto, bensi' o.99999999999999 come mostra il seguente esempio:

```
* R P N / 8 A1 (Vers.28.08.77)

:1 3/.3* FIX M PRINT END (h)
:(e)0.9999999999999
:
```

Soltanto una precisione "a infinite cifre significative" potrebbe dare il risultato corretto. Anche in queste condizioni comunque l'errore e' molto piccolo. Nel caso dell'esempio ora visto l'errore e' solo di 0.0000000000001 cioe' di un centomillesimo di miliardesimo.

Per la precisione delle funzioni trascendenti di cui dispongono certe versioni di RPN/8A si vedano i manuali di utenza.

APPENDICE A

Corrispondenze tasto/funzione per telescriventi in codice ASCII.

VARIABILI: @ A B C D E F G H I J K L M N O

NUMERI: 0 1 2 3 4 5 6 7 8 9

OPERATORI:

Tasto	Operatore	Eventuale nome
spazio	spazio	
!	GOTO	
"	"	Text
#	PRINT	
$	$	
%	END	
&	STO	
'		
(		
)		
*	*	Multiply
+	+	Sum
,	,	Separazione
-	-	Sub
.	.	Decimal Point
/	/	Divide